CLAIMS

What is claimed is:

1. A method for arithmetic overflow detection, comprising:

   receiving a first instruction defined for a first processor having a first base, said instruction comprising an operator and at least one operand having an operand type; and

   indicating whether said at least one operand has potential overflow based at least in part on said operator and the relationship between said operand type and a result type associated with said operator.

2. The method of claim 1 wherein said method further comprises converting said first instruction to a second instruction optimized for a second processor having a second base when said at least one operand does not have potential overflow, said second base smaller than said first base, said operand type belonging to said second base.

3. The method of claim 1, further comprising rejecting an expression that cannot be optimized to a smaller base on said second processor.

4. The method of claim 1 wherein said first instruction is arithmetic.

5. The method of claim 1 wherein said first instruction comprises a non-arithmetic, type-sensitive instruction.

6. The method of claim 2 wherein

said first processor comprises a Virtual Machine; and

said second processor comprises a Virtual Machine.

7. The method of claim 2 wherein

said first base is used by said first processor for performing arithmetic operations on at least one data type, said at least one data type having a size less than the size of said first base; and

said second base is used by said second processor for performing arithmetic operations on said at least one data type, said second base having a size equal to the size of said at least one data type.

8. The method of claim 2 wherein

said first processor comprises a 32-bit processor; and

said second processor comprises a resource-constrained 16-bit processor.

9. A method for arithmetic overflow detection, comprising:

step for receiving a first instruction defined for a first processor having a first base, said instruction comprising an operator and at least one operand having an operand type; and

step for indicating whether said at least one operand has potential overflow based at least in part on said operator and the relationship between said operand type and a result type associated with said operator.

10. The method of claim 9 wherein said method further comprises step for converting said first instruction to a second instruction optimized for a second processor having a second base when said at least one operand does not have potential overflow, said second base smaller than said first base, said operand type belonging to said second base.

11. The method of claim 9, further comprising step for rejecting an expression that cannot be optimized to a smaller base on said second processor.

12. The method of claim 9 wherein said first instruction is arithmetic.

13. The method of claim 9 wherein said first instruction comprises a non-arithmetic, type-sensitive instruction.

14. The method of claim 10 wherein

said first processor comprises a Virtual Machine; and

said second processor comprises a Virtual Machine.

15. The method of claim 10 wherein

said first base is used by said first processor for performing arithmetic operations on at least one data type, said at least one data type having a size less than the size of said first base; and

said second base is used by said second processor for performing arithmetic operations on said at least one data type, said second base having a size equal to the size of said at least one data type.

16. The method of claim 10 wherein

said first processor comprises a 32-bit processor; and

said second processor comprises a resource-constrained 16-bit processor.

17. A program storage device readable by a machine, embodying a program of instructions executable by the machine to perform a method for arithmetic overflow detection, the method comprising:

receiving a first instruction defined for a first processor having a first base, said instruction comprising an operator and at least one operand having an operand type; and

indicating whether said at least one operand has potential overflow based at least in part on said operator and the relationship between said operand type and a result type associated with said operator.

18. The program storage device of claim 17, said method further comprising converting said first instruction to a second instruction optimized for a second processor having a second base

when said at least one operand does not have potential overflow, said second base smaller than said first base, said operand type belonging to said second base.

19. The program storage device of claim 17, said method further comprising rejecting an expression that cannot be optimized to a smaller base on said second processor.

20. The program storage device of claim 17 wherein said first instruction is arithmetic.

21. The program storage device of claim 17 wherein said first instruction comprises a non-arithmetic, type-sensitive instruction.

22. The program storage device of claim 18 wherein

said first processor comprises a Virtual Machine; and

said second processor comprises a Virtual Machine.

23. The program storage device of claim 18 wherein

said first base is used by said first processor for performing arithmetic operations on at least one data type, said at least one data type having a size less than the size of said first base; and

said second base is used by said second processor for performing arithmetic operations on said at least one data type, said second base having a size equal to the size of said at least one data type.

24. The program storage device of claim 18 wherein

said first processor comprises a 32-bit processor; and

said second processor comprises a resource-constrained 16-bit processor.

25. An apparatus for arithmetic overflow detection, comprising:

means for receiving a first instruction defined for a first processor having a first base, said instruction comprising an operator and at least one operand having an operand type; and

means for indicating whether said at least one operand has potential overflow based at least in part on said operator and the relationship between said operand type and a result type associated with said operator.

26. The apparatus of claim 25 wherein said apparatus further comprises means for converting said first instruction to a second instruction optimized for a second processor having a second base when said at least one operand does not have potential overflow, said second base smaller than said first base, said operand type belonging to said second base.

27. The apparatus of claim 25, further comprising means for rejecting an expression that cannot be optimized to a smaller base on said second processor.

28. The apparatus of claim 25 wherein said first instruction is arithmetic.

29. The apparatus of claim 25 wherein said first instruction comprises a non-arithmetic, type-sensitive instruction.

30. The apparatus of claim 26 wherein

said first processor comprises a Virtual Machine; and

said second processor comprises a Virtual Machine.

31. The apparatus of claim 26 wherein

said first base is used by said first processor for performing arithmetic operations on at least one data type, said at least one data type having a size less than the size of said first base; and

said second base is used by said second processor for performing arithmetic operations on said at least one data type, said second base having a size equal to the size of said at least one data type.

32. The apparatus of claim 26 wherein

said first processor comprises a 32-bit processor; and

said second processor comprises a resource-constrained 16-bit processor.